UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/682,775	10/18/2001	Jason J. Harms	2290	4210

24333 7590 10/23/2006

GATEWAY, INC.
ATTN: Patent Attorney
610 GATEWAY DRIVE
MAIL DROP Y-04
N. SIOUX CITY, SD 57049

EXAMINER	
VO, TED T	

ART UNIT	PAPER NUMBER
2191	

DATE MAILED: 10/23/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/682,775	HARMS, JASON J.	
	Examiner	Art Unit	
	Ted T. Vo	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *27 July 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-26 and 29-32* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-26, 29-32* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the amendment filed on 07/27/2006.

Claims 27-28 are canceled. Claims 1-26, 29-32 are pending.

Terms' Definition

2. From Wikipedia, the free encyclopedia:

* **Configuration files**: (http://en.wikipedia.org/wiki/Configuration_file)

In computing, configuration **files**, or config files, are used to configure the initial settings for some computer programs. They are used for user applications, server processes and operating system settings. The files are often written in ASCII (rarely UTF-8) and line-oriented, with lines terminated by a newline or carriage return/line feed pair, depending on the operating system. They may be considered a simple database. Some files are created and modified using an ASCII editor. Others are created and modified as a side-effect of changing settings in a graphical user interface (GUI) program. The formats of configuration files are often poorly documented.

Some applications provide tools to create, modify, and verify the syntax of their configuration files. For server processes and operating system settings, the only documentation may be the source code. Some configuration files are partially described by man or help pages.

Some computer programs only read the configuration files at startup. Others periodically check the configuration files for changes. Some can be told to re-read the configuration files and apply the changes to the current process. There are no standards or strong conventions.

Recently, XML and YAML have become popular as configuration file formats. They have the advantages of having well-defined syntaxes, and tools to validate and verify the syntax of the files that are created in those formats.

Microsoft Windows

Within the Microsoft Windows family of operating systems and their attendant applications, the situation is similar. Windows 3.0 had an API for INI files (from "initialization"), but that format is deprecated and many modern Windows programs forgo configuration files to use only the Windows Registry to store information.

* **INI Files** (http://en.wikipedia.org/wiki/Initialization_file)

An **initialization file**, or **INI file**, is a configuration file that contains configuration data for Microsoft Windows based applications. Starting with Windows 95, the INI file format was superseded but not entirely replaced by a registry database in Microsoft operating systems.

* **Windows registry** (http://en.wikipedia.org/wiki/Windows_Registry).

In computing, the **Windows registry** is a database which stores settings and options for the operating system for Microsoft Windows 32-bit versions, 64-bit versions and Windows Mobile. It contains information and settings for all the hardware, software, users, and preferences of the PC. Whenever a user makes changes to "Control Panel" settings, or file associations, system policies, or installed software, the changes are reflected and stored in the registry.

The Windows Registry was introduced to tidy up the profusion of per-program INI files that had previously been used to store configuration settings for Windows programs. These files tended to be scattered all over the system, which made them difficult to keep track of.

Contents

[hide]

- 1 Registry structure
  - 1.1 HKEY_CLASSES_ROOT
  - 1.2 HKEY_CURRENT_USER
  - 1.3 HKEY_LOCAL_MACHINE
  - 1.4 HKEY_USERS
  - 1.5 HKEY_CURRENT_CONFIG
- 2 Editing the Registry
  - 2.1 Manual editing
  - 2.2 Command line editing
  - 2.3 Code editing
- 3 Where is the Registry stored?
  - 3.1 Windows NT, 2000, XP, and Server 2003
  - 3.2 Windows 95, 98, and Me
  - 3.3 Windows 3.11
- 4 Policy files
  - 4.1 Policy file editor
- 5 Useful Registry keys
- 6 Utilities
- 7 Advantages of the Registry concept
- 8 Criticisms of the Registry concept
- 9 Registry alternatives in other operating systems
- 10 Problems with Windows 9x OS
- 11 See also
- 12 References

- 13 External links

[edit]

Registry structure
The Registry is split into a number of logical sections. These are generally known by the names of the definitions used to access them in the Windows API, which all begin "HKEY" (an abbreviation for "Handle to Key"); often, they are abbreviated to a three- or four-letter short name starting with "HK".
Each of these keys is divided into subkeys, which may contain further subkeys, and so on. Any key may contain values. These values can be:
String Value
Binary Value (0 and 1's)
DWORD Value (numbers between 0 and 4,294,967,295 [$2^{32} - 1$])
Multi-String value
Expandable String Value
Each key has a default value, which is in effect a value with the same name as the key. Registry keys and values are specified with a syntax similar to Windows' filenames, using backslashes to indicate levels of hierarchy. E.g. HKEY_LOCAL_MACHINE\Software\Microsoft\Windows refers to the subkey "Windows" of the subkey "Microsoft" of the subkey "Software" of the HKEY_LOCAL_MACHINE key.
The HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER nodes have a similar structure to each other; applications typically look up their settings by first checking for them in "HKEY_CURRENT_USER\Software\Vendor's name\Application's name\Version\Setting name", and if the setting is not found looking instead in the same location under the HKEY_LOCAL_MACHINE key. When writing settings back, the reverse approach is used — HKEY_LOCAL_MACHINE is written first, but if that cannot be written to (which is usually the case if the logged in user is not an administrator), the setting is stored in HKEY_CURRENT_USER instead.
[edit]

HKEY_CLASSES_ROOT

Abbreviated HKCR, HKEY_CLASSES_ROOT stores information about registered applications, including associations from file extensions and OLE object class ids to the applications used to handle these items. On Windows 2000 and above, HKCR is a compilation of HKCU\Software\Classes and HKLM\Software\Classes. If a given value exists in both of the subkeys above, the one in HKCU\Software\Classes is used.
[edit]

HKEY_CURRENT_USER

Abbreviated HKCU, HKEY_CURRENT_USER stores settings that are specific to the currently logged in user. HKCU mirrors the current user's subkey of HKEY_USERS.
[edit]

HKEY_LOCAL_MACHINE

Abbreviated HKLM, HKEY_LOCAL_MACHINE stores settings that are general to all users on the computer. This key is found within the file %SystemRoot%\System32\Config\system on NT-based versions of Windows. Information about system hardware is located under the SYSTEM key.
[edit]

HKEY_USERS

Abbreviated HKU, HKEY_USERS contains subkeys corresponding to the HKEY_CURRENT_USER keys for each user registered on the machine.
[edit]

HKEY_CURRENT_CONFIG

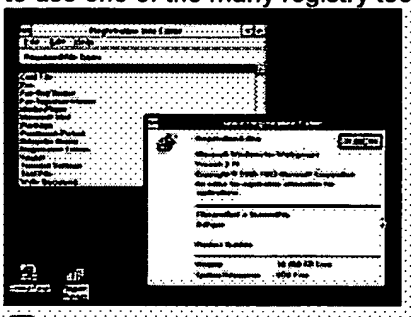
Abbreviated HKCC, HKEY_CURRENT_CONFIG contains information gathered at runtime; information stored in this key is not permanently stored on disk, but rather regenerated at boot time.
[edit]

Editing the Registry

[edit]

Manual editing

The registry can be edited manually in Microsoft Windows by running regedit.exe or regedt32.exe in the Windows directory. However, careless registry editing can cause irreversible damage. Many optimization and "hacking" tools are available to modify this portion of the Windows operating system. It is preferable to use one of the many registry tools available.



Windows 3.11 Registration Editor
A simple implementation of the current registry tool appeared in Windows 3.x, called the "Registration Info Editor" or "Registration Editor". This was basically just a database of applications used to edit embedded OLE objects in documents.
Windows NT introduced permissions for Registry editing. Windows NT 4 and Windows 2000 were distributed with both the Windows 9x REGEDIT.EXE program and Windows NT 3.x's REGEDT32.EXE program. There are several differences between the two editors on these platforms:

- REGEDIT.EXE had a left-side tree view that began at "My Computer" and listed all loaded hives. REGEDT32.EXE had a left-side tree view, but each hive had its own window, so the tree displayed only keys.
- REGEDIT.EXE represented the three components of a value (its name, type, and data) as separate columns of a table. REGEDT32.EXE represented them as a list of strings.
- REGEDIT.EXE supported right-clicking of entries in a tree view to adjust properties and other settings. REGEDT32.EXE required all actions to be performed from the top menu bar.
- Because REGEDIT.EXE was directly ported from Windows 95, it did not support permission editing (permissions do not exist on Windows 9x). Therefore, the only way to access the full functionality of an NT registry was with REGEDT32.EXE.
- REGEDIT.EXE only supported string (REG_SZ), binary (REG_BINARY), and DWORD (REG_DWORD) values. REGEDT32.EXE supports those, plus expandable string (REG_EXPAND_SZ) and multi-string (REG_MULTI_SZ). **Attempting to edit unsupported key types with REGEDIT.EXE on Windows 2000 or Windows NT 4 will result in registry corruption and, possibly, an unbootable system.**[1]

Windows XP was the first system to integrate these two programs into one, adopting the old REGEDIT.EXE interface and adding the REGEDT32.EXE functionality. The differences listed above are not applicable on Windows XP and newer systems; REGEDIT.EXE is the improved editor, and REGEDT32.EXE simply invokes REGEDIT.EXE.
[edit]

Command line editing

On NT-based systems the registry can be manipulated from the command line with the reg.exe utility. It is included in Windows XP and can be downloaded separately for previous versions.
reg.exe Operation [Parameter List]
Operation
[QUERY|ADD|DELETE|COPY|SAVE|LOAD|UNLOAD|RESTORE|COMPARE|EXPORT|IMPORT]
Also, a .reg file (a text-based human-readable file format for storing portions of the registry) can be imported from the command line with the following command:
regedit.exe /s *file*
The /s means the file will be *silent merged* to the Registry. If the /s parameter is omitted the user will be asked to confirm the operation. In Windows 98 and Windows 95 the /s switch also caused regedit.exe to ignore the setting in the registry that allows administrators to disable it. When using the /s switch Regedit does not return an appropriate return code if the operation fails, unlike reg.exe which does. This makes it hard to script, however a possible workaround is to add the following lines into your batch file:

```
regedit /s file.reg
regedit /e test.reg "key"
if not exist test.reg goto REGERROR
del test.reg
```

The default association for .reg files in many versions of Microsoft Windows, starting with Windows 98 does require the user to confirm the merging to avoid user mistake.
[edit]

Code editing

You can edit the registry through the APIs of the Advanced Windows 32 Base API Library (advapi32.dll) [1], [2].
This is a list of the Registry API Functions:

RegCloseKey	RegOpenKey
RegConnectRegistry	RegOpenKeyEx
RegCreateKey	RegQueryInfoKey
RegCreateKeyEx	RegQueryMultipleValues
RegDeleteKey	RegQueryValue
RegDeleteValue	RegQueryValueEx
RegEnumKey	RegReplaceKey
RegEnumKeyEx	RegRestoreKey
RegEnumValue	RegSaveKey
RegFlushKey	RegSetKeySecurity
RegGetKeySecurity	RegSetValue
RegLoadKey	RegSetValueEx
RegNotifyChangeKeyValue	RegUnLoadKey

Another way is to use the Windows Support Tool Reg.exe by executing it from your code [3]
[edit]

Where is the Registry stored?

The Registry is stored in several files; depending upon the version of Windows, there will be different files and different locations for these files, but they are all on the local machine, except for the NTuser or user file which may be placed on another computer to allow for roaming profiles.
[edit]

Windows NT, 2000, XP, and Server 2003

The following Registry files are stored in %SystemRoot%\System32\Config\:
Sam - HKEY_LOCAL_MACHINE\SAM
Security - HKEY_LOCAL_MACHINE\SECURITY
Software - HKEY_LOCAL_MACHINE\SOFTWARE
System - HKEY_LOCAL_MACHINE\SYSTEM
Default - HKEY_USERS\DEFAULT
Userdiff
The following file is stored in each user's profile folder:
NTUSER.dat
[edit]

Windows 95, 98, and Me

The registry files are named User.dat and System.dat and are stored in the \Windows\ directory. In Windows Me Classes.dat was added.
[edit]

Windows 3.11

The registry file is called Reg.dat and is stored in the \Windows\ directory.
[edit]

Policy files

Since Windows 95, administrators can use a special file to be merged into the registry, a policy file. The policy file allows administrators to enforce registry settings such as preventing users from changing the background picture of the desktop. The default extension for the policy file is .pol. The policy file filters the settings it enforces on a per user basis and per user group basis. To do that the policy file merges into the registry, preventing users from circumventing it by simply changing back the settings. The policy file is usually distributed through a LAN, but can be placed on the local computer.
[edit]

Policy file editor

The policy file is created by a free tool by Microsoft that goes by the filename poledit.exe for Windows 95/Windows 98 and with a computer management module for NT-based systems. The module will not work in Windows XP Home Edition, but it does work in the Professional edition. The editor requires administrative permissions to be run on systems that uses permissions. The editor can also directly change the current registry settings of the local computer and if the remote registry service is installed and started on another computer it can also change the registry on that computer. The policy editor loads the settings it can change from .adm files, of which one is included, that contains the settings the Windows shell provides. The .adm file is plain text and supports easy localisation by allowing all the

strings to be stored in one place. The policy editor has been replaced by Group Policies in newer versions of Windows.
[edit]

Useful Registry keys

The following registry keys may be of interest to users attempting to customize their Windows systems.

- **HKLM\System\CurrentControlSet\Control\FileSystem\NtfsDisableLastAccessUpdate** Creating this (as a DWORD) and setting it to 1 will prevent Windows (NT, 2000 or XP) from tracking the last access time of files, which speeds up a lot of operations (especially opening folders of items with previews).
- **HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters\SizReqBuf** Specifies the size of buffers used for storing requests to the file/print server. Increasing this from the default of 4356 bytes can improve network performance: a figure of 14596 is frequently recommended.
- **HKLM\Software\Microsoft\Windows\CurrentVersion\Run** (and the **HKCU** equivalent) specifies applications to run whenever a user logs in. These can include desirable programs, such as printer monitoring programs or frequently-used tools, but a lot of malware uses this registry key to ensure it is automatically run. This key is a good place to start looking for evidence of malware if you think your computer has been infected.

[edit]

Utilities

- Registry inspection and monitoring tools
  - Regmon; part of the Sysinternals tools — A tool for detailed monitoring of applications that are accessing registry items
  - Registry Jumper — A freeware utility for quick access to registry keys
  - CCleaner — A system optimization tool that, among other things, detects and corrects application-related registry problems
  - Regclean — An unsupported tool, originally from Microsoft, to remove old Registry entries
  - Regvac Registry Cleaner — A system optimization utility for registry cleaning
  - RegSeeker — A system tuning utility that includes registry cleaning
  - NTREGOPT — A registry optimizer for Windows NT/2000/2003/XP
  - jv16 PowerTools — An utility suite containing a registry cleaner, a registry monitor and a registry compactor.
  - RegCompact.NET — A freeware registry compactor for Windows NT/2000/2003/XP
  - Chntpw — An opensource offline Windows Registry/SAM editor that runs under Linux
  - ERD Commander — A bootable CD which includes an off-line registry editor for repairing Windows installations.

- Registry Workshop — A much-improved registry editor, intended to replace regedit.exe/regedit32.exe for general use.

A review of various Registry cleaners was carried out by Fred Langa and published in *Information Week* on October 10, 2005.
[edit]

Advantages of the Registry concept

Changing from having one or more INI files per program to one centralised registry has its good points:

- The registry keeps machine configuration separate from user configuration. When a user logs into a Windows NT/2000/XP/Server 2003 computer, his or her user-based registry settings are loaded from a different path than the system wide settings. This allows programs to more easily keep per-user configuration, as they can just work with the "current user" key, whereas in the past they tended to just keep system-wide per-program settings. (This point doesn't apply to programs on Unix/Linux-based OSes as they have an accepted standard for per-user settings, where Windows previously did not).
- Group Policy allows administrators on a Windows-based computer network to centrally manage program and policy settings. Part of this involves being able to set what an entry in the registry will be for all the computers on the network, and affect nearly any installed program — something almost impossible with per-program configuration files each with custom layouts, stored in dispersed locations.
- Because the registry is accessed through a special API it is available to scripts and remote management using WMI. Each script does not have to be customised for every application's unique configuration file layouts and restrictions.
- The registry can be accessed as one item over a network connection for remote management/support, including from scripts, using the standard API.
- It can be backed up more easily, in that it is just a small number of files in specific locations.
- Portions of settings like any subset of an application configuration can be saved in a text-based .REG file, which can be edited with any text editor later. .REG files can easily be merged back into the registry both by unattended batch file or by the user using just a double-click on the file without harming any setting that is not explicitely stated in the .REG file. This is very useful for administrators and support personel which want to preset or preconfigure only a few options like approving the EULA of Acrobat Reader.
- Since accessing the registry does not require parsing it can be read from and written to more quickly than a text file can be.
- Registry changes and readings can be tracked via a tool like SysInternals RegMon on value level. This is a big advantage for generating scripts in networks as well as debugging problems.
- Registry keys are independent of the Windows language, the Windows installation drive and path and even the Windows versions as such. So support personel can easily give out one set of instructions, without having to handle these things, unlike for example files in the user profile which can be on different paths on each installation.

[edit]

Criticisms of the Registry concept

However, the centralized Registry introduces some problems as well:

- The HKEY_LOCAL_MACHINE part is a single point of failure — damage to the Registry can render a Windows system unbootable, in extreme cases to a point that can not be fixed, and requires a full reinstall of Windows. There is an automated backup mechanism however, these secondary files which will be loaded, if the primary files fail to load.
- Any program which wants to manipulate the registry must use special Windows API functions, rather than a simple text configuration file. This is an issue when writing applications designed for use on several platforms.
- The registry does not document itself in the same way a configuration file can.
- Restoring parts of the registry is hard because you cannot easily extract data from backed up registry files. Offline reading and manipulation of the registry (for example from a parallel installed Windows or a boot cd) is not trivial (but not impossible).
- Any application that doesn't uninstall properly, or doesn't have an uninstaller, can leave entries in the registry. In rare cases this can lead to performance or even stability problems, but only if the application registers itself as a class in HKEY_CLASSES_ROOT. (Note that user settings usually remain in the registry, which is done by design for two reasons: first, the user might be on a Windows domain with server-based profiles, where the settings move with the user to other computers. Uninstalling the application on one computer does not mean the user does not want to use the program on some other computer on the domain. Second, the uninstall process would only be able to modify the current user's settings anyway. In any case, unused keys in HKCU have negligible impact on system performance).
- Since at least 1998 [4], pages on Microsoft's support website relating to editing the registry include the disclaimer "Use Registry Editor at your own risk." (see, for example, [5]) This curt disclaimer might imply that Microsoft's tech support are unable to predict the effects of certain registry edits on the overall operation of Windows and its client applications. It might also imply an attempt to limit Microsoft's legal liability in instances where registry edits caused damage (e.g. lost productivity, corrupted files, etc.) to a customer's machine.

[edit]

Registry alternatives in other operating systems

Other systems preserve the concept of separate configuration files for separate application subsystems, but group them together in a single filesystem directory for ease of management, such as the /etc and hidden directories (directories that start with a period) within the home directory in Unix-like systems. In those systems, fine-grained access to configuration settings can be controlled by normal filesystem protection mechanisms. Also, the only thing that could cause widespread damage to the configuration system would have to be major filesystem corruption.
Mac OS X has most application settings stored as property list files which are usually stored in each user's Library folder. This means that uninstalling an application by deleting the application bundle can leave unused property list files on the computer, much like registry entries on Windows. The biggest difference is that each application's settings are stored in separate files so that damage to settings files normally affect only individual applications.
RISC OS also allows applications to be copied into directories easily without the need to install the application as one would in Windows, if one wishes to remove the application, simply delete the folder

belonging to the application [6]. This is possible, since RISC OS does not support multi-user environments with different settings for each user.
[edit]

Problems with Windows 9x OS

On Windows 9x computers, an older installation can have a very large registry that slows down the computer's startup and can make the computer unstable. This has led to frequent criticisms that the registry leads to instability. However, these problems occur far less often on the Windows NT family of systems, including Windows XP.

Response to Arguments

3. Applicant's arguments with respect to the amended claims filed on 07/27/2006 have been fully considered.

With regards to the arguments/amendments, "automatically without user input", it should be further noted to Applicants:

> In re Venner, 262 F.2d 91, 95, 120 USPQ 193, 194 (CCPA 1958) (Appellant argued that claims to a permanent mold casting apparatus for molding trunk pistons were allowable over the prior art because the claimed invention combined "old permanent-mold structures together with a timer and solenoid which automatically actuates the known pressure valve system to release the inner core after a predetermined time has elapsed." **The court held that broadly providing an automatic or mechanical means to replace a manual activity** which **accomplished the same result** is **not sufficient** to distinguish over the prior art.). (Emphasis added)
> (See MPEP 2144.04).

(And this was addressed in the prior office action).

The adding limitation, "automatically without user input", clearly preempts a common manual act on the Microsoft Windows' registry, where the use of a program to do for a manual act is not new in the art.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

5. Claims 1-16 are rejected under 35 U.S.C. 102(b) as being anticipated by an Microsoft System Journal (Hereinafter: Microsoft), "Under The Hood", 9-1996.

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per Claim 1: Microsoft discloses using a program to scan a registry of windows, as Microsoft said, deleting a file simply search through the file name and registry path and deletes (see text under "The CLENREG User Interface"). However, the problem is that the dangling still remains under the subkeys in the registry. Microsoft discloses scanning string searching a key and all subkeys relating to a dead file, and then cleaning such key/subkeys (see p. 2, start at "ScanRegNode"). Thus, Microsoft discloses the below limitation having the functionality as shown by Microsoft:

A method for automatically removing entry of a device from a computer system identified by the system as not being property identified, said method comprising:

scanning configuration data using executable computer code to determined an entry for a device with not properly identified by the system; and

removing automatically, without user input, the entry for the device from the configuration data after determining the device is not properly identified . Further see pages:2-6.

Note: with regard to: *without user input,* see MPEP 2144.04: " The court held that broadly providing an automatic or mechanical means to replace a manual activity which accomplished the same result is not sufficient to distinguish over the prior art).

As per Claim 2: Windows 98 discloses, *The method of claim 1, further comprising:*

determining a vendor of the device (i.e., the file name that has registered in root of the registry path, provided within the registry and Microsoft wants to delete. E.g., "MICROSOFT" in the registry as shown in Figure 2, is a vendor);

scanning all subkeys in the configuration data for all devices associated with the vendor; and deleting all keys associated with the devices associated with the vendor automatically without user input after the subkeys associated with the vendor have been located during said scanning.

Refer as "Search and Delete Call Trees for CLEANREG. See pages: 2-6.

Note: with regard to: *without user input,* see MPEP 2144.04: " The court held that broadly providing an automatic or mechanical means to replace a manual activity which accomplished the same result is not sufficient to distinguish over the prior art).

As per Claim 3: Microsoft discloses, The method of claim 1, further comprising: modifying an initialization file to remove device information automatically by computer system without manual user input.

With regard to "initialization file", see definition of Windows Registry above:

> The Windows Registry was introduced to tidy up the profusion of per-program INI files that had previously been used to store configuration settings for Windows programs.

Therefore modifying of registry is to modify a configuration file, an INI file. Accordingly, the reference teaches the modification using a program code, i.e., without manual user input.

Furthermore, based on the claimed context, it should be noted that after an initialization has been modified, i.e., changed, then when the computer system reads it, the system does automatically without manual user input.

Also, see definition of Configuration file as listed above. A Standard Microsoft Windows requires configuration files or initialization files as a part of Window configuration system. The Windows discloses this feature – Note Wikipedia defines an **initialization file**, or **INI file**, is a configuration file that contains

configuration data for Microsoft Windows based applications. Starting with Windows 95, the INI file format was superseded but not entirely replaced by a registry database in Microsoft operating systems. Although made popular by Windows, INI files can be used on any system thanks to their flexibility. They allow a program to store configuration data, which can then be easily parsed and changed).

Further note: with regard to*: without user input,* see MPEP 2144.04: " The court held that broadly providing an automatic or mechanical means to replace a manual activity which accomplished the same result is not sufficient to distinguish over the prior art).

As per Claim 4: Microsoft discloses, *The method of claim 1, further comprising: deleting files identified in a file list* (See CLEANREG does it). It should be noted that, the claim recites: *further comprising: deleting files identified in a file list.* It is clearly that this limitation covers all acts of deleting a file/files in a computer, without depending acts from the claim 1, Microsoft Windows provides "DELETE"/"CUT" where a user positions a mouse on a file icon, or user drags a file list, the command will delete the file/files.

As per Claim 5: Microsoft discloses, *The method of claim 4, wherein the deleting element further comprises: saving a backup copy of the files prior to deletion automatically without manual user input.* Claim 5 is depending on claim 4, where claim 4 recites deleing a file/files that clearly cover independently from the step done in claim 1, in a Microsoft Windows, Windows commands such as, SAVE, SAVE AS, provide save the file including it backup automatically. When position on the file, Windows' commands such as, DELETE and CUT perform file deleting, The Windows discloses this claiming. Furthermore, when deleting a file in a Microsoft Windows, this file does not go, it is stored in a trash where this act does not need a user input.

Note: with regard to*: without user input,* see MPEP 2144.04: " The court held that broadly providing an automatic or mechanical means to replace a manual activity which accomplished the same result is not sufficient to distinguish over the prior art).

As per Claim 6: Regarding limitation of Claim 6, see the rationale discussed in the rejection of Claim 1.

As per Claim 7: Regarding limitation of Claim 7, see the rationale discussed in the rejection of Claim 2.

As per Claim 8: Regarding limitation of Claim 8, see the rationale discussed in the rejection of Claim 3.

As per Claim 9: Regarding limitation of Claim 9, see the rationale discussed in the rejection of Claim 4.

As per Claim 10: Regarding limitation of Claim 10, see the rationale discussed in the rejection of Claim 1.

As per Claim 11: Regarding limitation of Claim 11, see the rationale discussed in the rejection of Claim 2.

As per Claim 12: Regarding limitation of Claim 12, see the rationale discussed in the rejection of Claim 4.

As per Claim 13: Regarding limitation of Claim 13, see the rationale discussed in the rejection of Claim 1.

As per Claim 14: Regarding limitation of Claim 14, see the rationale discussed in the rejection of Claim 2.

As per Claim 15: Regarding limitation of Claim 15, see the rationale discussed in the rejection of Claim 2.

As per Claim 16: Regarding limitation of Claim 16, see the rationale discussed in the rejection of Claim 3.

6. Claims 17-21, 23-26, 29-32 are rejected under 35 U.S.C. 102(b) as being anticipated by an online reference, "Windows 98 – UBS TROUBLESHOOTING REFERENCE" (Hereinafter: Windows 98).

As per Claim 17: Regarding limitation of Claim 17, Windows 98, providing manual instructions, the instructions disclose the limitation of Claim 17:

"*means for removing a registry key associated with a predetermined device of a computer system without a user manually searching for the registry key*" (see p.1 "a USB device not being recognized", see p. 4, USB DETECTION and INSTALLATION STEPS: "The wizard will search for a USB Human Interface device", and see p. 6, "WHAT THE REGISTRY MAY LOOK LIKE");

"*and means for modifying a configuration file to indicate removal of the predetermined device from the computer system without a user manually modifying the configuration file* (See p. 2, e.g. text inside USB Mouse CONFIGURATION, and see p. 5, "when the registry is modified for the USB mouse".

> Note: see definition of Windows Registry: The Windows Registry was introduced to tidy up the profusion of per-program INI files that had previously been used to store configuration settings for Windows programs.
>
> Note: with regard to: *without user input,* see MPEP 2144.04: " The court held that broadly providing an automatic or mechanical means to replace a manual activity which accomplished the same result is not sufficient to distinguish over the prior art).

"*where the predetermined device is removed from the computer system so as to not interference with a subsequent installation*" (See p. 5-6, particularly, see Locate the key, and see Hit Delete to remove the device node).

As per Claim 18: Regarding limitation of Claim 18, refer to remove/delete discussed in the reference.

As per Claim 19: Regarding limitation of Claim 19, it discloses Microsoft, HKEY_LOCAL_MACHINE\ .."all subkeys"...\. See Microsoft registry structure, appeared in the reference; particularly as noted in the definition above.

As per Claim 20: Regarding limitation of Claim 20, it discloses Windows 98 files, see p.2, "WINDOWS\SYSTEM32\...".

As per Claim 21: Regarding limitation of Claim 21, it discloses Windows 98 files, see p.6, "HKEY_LOCAL_MACHINE\enum\...".

As per Claim 23: Windows 98 discloses Registry editor that can perform removing means such as performed on an information storage medium and load on to a computer (see p. 5: Insert Disk, and p.6).

As per Claim 24: Windows 98 discloses Registry editor that can perform configuring means such as performed on an information storage medium and load on to a computer (see p. 5: Insert Disk, and see p.1, the steps).

As per Claim 25: Windows 98 discloses Registry editor that can perform configuring means such as performed on an information storage medium and load on to a computer (see p. 5: Insert Disk, and see p. 1, the steps).

As per Claim 26: Windows 98 discloses Registry editor that can perform modifying means such as performed on an information storage medium and load on to a computer (see p. 5: Insert Disk, and see p. 5, the text under "WHAT THE REGISTRY MAY LOOK LIKE").

As per Claim 29: Windows 98 discloses,

Where the device is considered to be not properly identified by the system when

(i) not identified by the system,

(ii) not completely recognized by the system, or

(iii) only identified as a generic device by the system.,

See top paragraph in p. 1.

As per Claim 30: Claim 30 is indefinite as noted: For the limitation recited in claim 30: Windows 98 discloses this limitation. See top paragraph in p. 1.

As per Claim 31: See rationale in the rejection of Claim 29.

As per Claim 32: See rationale in the rejection of Claim 29.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. Claim 22 is rejected under 35 U.S.C. 103(a) as being unpatentable over an online reference, last modified by www.targus.com/us/ on 05/30/00, "Windows 98 – UBS TROUBLESHOOTING REFERNCE" (hereinafter: Windows 98) .

As per claim 22: Win.ini file is an element of WINDOWS 98 that has therein predetermined devices used/associated with the Windows system registry.

Windows 98 does not explicitly shown in the WIN.INI the predetermined device such as: load=, run=, device=lines, as claimed in the Claim 22.

However these differences are only found in the nonfunctional descriptive material and are not functionally involved in the steps/means recited. The predetermined device in the WIN.INI in Microsoft Windows would be performed the same regardless of the type of device data as claimed. Thus, this descriptive material will not distinguish the claimed invention from the prior art in terms of patentability, *see In re Gulack*, 703 F.2d 1381, 1385, 217 USPQ 401, 404 (Fed. Cir. 1983); In *re Lowry*, 32 F.3d 1579, 32 USPQ2d 1031 (Fed. Cir. 1994).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to clear the data as shown in the claim 22 as the type of data or information as for

conforming to the uninstalled requirements of the Windows, and because such data or information does not functionally relate to the steps in the method claimed and because the subjective interpretation of the data does not patentably distinguish the claimed invention.

Conclusion

9. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 4:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for

unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV
October 13, 2006

TED VO
PRIMARY EXAMINER
TECHNOLOGY CENTER 2100